DOCUMENT RESUME

ED 052 600 EM 009 046

AUTHOR	Bunderson, C. Victor
TITLE	Current Issues in the United States Regarding CAI.
INSTITUTION	Texas Univ., Austin. Computer-Assisted Instruction Lab.
SPONS AGENCY	Department of Defense, Washington, D.C. Advanced Research Projects Agency.; National Science Foundation, Washington, D.C.
REPORT NO	TM-3
PUB DATE	Feb 70
NOTE	43p.; Speech given at the Conference on Computers and Education, University of Bari, Italy, October, 1970
EDRS PRICE	EDRS Price MF-$0.65 HC-$3.29
DESCRIPTORS	*Computer Assisted Instruction, Curriculum Design, *Curriculum Development, Discovery Learning, *Instructional Design, *Instructional Systems, Learning Processes, Teacher Developed Materials

ABSTRACT

Two issues inhibit full utilization of computer-assisted instruction (CAI). The first of these issues concerns authorship of CAI programs. Should it be left with classroom teachers or given to specially trained instructional design teams? The latter approach is preferable for "mainline" systems (those systems that are specifically designed to teach a complete course), and the former is better for "adjunct" systems (those that supplement a teacher's regular course). Instructional software design teams can provide a full range of information, such as documentation and justification, a management plan for development, and heuristics for quality control that are useful for supplemental programs but almost indispensable for full-scale or mainline systems. The second issue concerns whether CAI should emphasize discovery learning or carefully controlled programs (expository instruction). Studies have shown that bright students do better with discovery learning, but that average or below average students learn more from expository instruction. For slower students, programs may use discovery techniques within the context of more carefully structured, ordered programs. (JK)

CURRENT ISSUES IN THE UNITED STATES
REGARDING CAI

Technical Memo No. 3

C. Victor Bunderson

1

ED052600

CURRENT ISSUES IN THE UNITED STATES REGARDING CAI

Technical Memo No. 3

C. Victor Bunderson

February 1970

Supported By:

THE NATIONAL SCIENCE FOUNDATION
Grant GJ 509 X

and

ADVANCED RESEARCH PROJECTS AGENCY
Contract N00014-67-0126-0006

The University of Texas at Austin
Computer-Assisted Instruction Laboratory
C. Victor Bunderson, Director
Austin, Texas 78712

CURRENT ISSUES IN THE UNITED STATES REGARDING CAI[1,2]

C. Victor Bunderson
Director

Computer-Assisted Instruction Laboratory
The University of Texas at Austin
Austin, Texas 78712

The promise of computer-assisted instruction (CAI) has been widely heralded, but slow to coming to fruition. As with other technological innovations which have about them the prospects of revolution, inevitable conflict with strongly held attitudes, values, and habits produces much heat but often little light. In the author's experience, many of the disagreements regarding the use of CAI in higher education can be related to two fundamental issues. It is the purpose of this paper to describe these two issues and to provide a context which can largely make the two issues irrelevant. This new context is provided by a distinction between "adjunct" and "mainline" CAI applications and through the partial definition of a new discipline of "instructional software design."

The first issue regards the authorship of CAI programs. Should it be left with the classroom teachers or given to specially-trained instructional design teams? The second issue could be characterized as "programmed instruction" vs. free and flexible man-machine problem solving. As will be seen, these two issues have important aspects in common.

---

[1]Address presented at the Conference on Computers and Education, University of Bari, Italy, October, 1970.

[2]The research reported in this paper was supported in part by the Advanced Research Projects Agency, Department of Defense, and was monitored by the Office of Naval Research, Contract N00014-67-A-0126-0006, and the National Science Foundation, Washington, D. C., Grant GJ 509 X.

The first issue takes on many faces in debates in the United States. Should classroom teachers or professors perform CAI program development themselves? Can they do it themselves? Or should highly-trained "instructional software designers" gain important authority in the development process? Other questions reverberate from this issue. Should the classroom as a unit be replaced by resource centers and work areas for individualized self-study or small group study? Should the role of "teacher" be fractionated into differential staffing patterns, including those of instructional designer and instructional manager? Will control of the content of education pass to centralized decision-making bodies? One way of characterizing this issue is: Should CAI be viewed as new media for the teacher or as a new technology which redefines the teacher and the classroom?

On this issue, I take the position that CAI should be viewed as a new technology and not as another medium of instruction for the individual classroom teacher. As such, it will require extensive research and development which will lead, in part, to the definition of a new discipline of *Instructional Software Design*. This discipline will prove to be a critical component in the full development of computer uses in education. It will be a synthesis and extension of parts of several existing fields:

> *Instructional* implies education disciplines, especially educational psychology, involving human learning, psychometrics, programmed instruction, and individual differences.

Software implies computer science subdisciplines, especially system analysis, programming, artificial intelligence, natural language processing, and computer graphics.

Design implies a problem-oriented engineering approach--an iterative empirically oriented pragmatic approach rather than a theory-oriented "natural science" approach.

Simon (1969) would characterize this new technology as an "artificial science" because of this last-named fact. In the book cited, he has laid the outlines for a science of design, applicable across many disciplines. Instructional Software Design, or more briefly, "instructional design," is a branch of that new "science of the artificial," not mentioned by Simon, but definitely in the spirit of his "Design Science" approach.

Instructional Software Design, then, is the technology of instructional materials, their design and development, and the future instructional theory behind this technology. As much as the hardware, it has the potential of changing education from the labor-intensive, low-yield field it now is to a man-machine system possessing probably a lower "teacher"-pupil ratio but with a far higher yield in terms of educational accomplishments and values on the part of students. The analogy to agriculture is relevant. Agriculture was transformed by technology and by a system of agricultural agents from a labor-intensive, low-yield system to the highly-efficient and productive technology-intensive field known today in modern countries.

Such a transformation cannot come as long as computers remain on the periphery of education. If classroom teachers are left the full

responsibility for development, the result will be primarily supplements to traditional classroom or laboratory work, homework using the computer, etc. These computer applications can have an important impact on education, but they should be viewed as complements to computer applications which take on a mainline instructional burden.

Figure 1 illustrates a useful distinction between "adjunct" and "mainline" uses of computers in education which helps place the first issue in context. By adjunct is meant those applications of computers used by teachers as supplements to their regularly scheduled (semester or quarter) courses. These uses include problem solving using the computer, simulation and modeling, use of programs for illustration during lectures, and the use of subroutines or macros to generate drill, practice, or similarly constrained exercise material. At the State University of New York at Stony Brook, extensive exercise material in foreign language instruction has been generated by this approach (Morrison & Adams, 1969; Adams, 1969), by eliminating the slow programming, coding, and debugging process. In the case of these language materials, it is seen that enough material can be generated to automate a large portion of a language course. The fact that adjunct program development can lead to "mainline" use of computers is recognized in Figure 1 by the arrows leading from the adjunct end of the continuum to the mainline end. If enough single-concept films are produced over a period of time, they could similarly be organized into a teaching system.

Insert Figure 1 about here

A "mainline" CAI application differs primarily in that it is a complete system to teach, at the least, a complete course to students from a definite population having certain defined prerequisites. Considering CAI as a new technology rather than as another kind of medium to be used by the teacher in his traditional classroom has important consequences in terms of staffing, costs and effectiveness, and the organization and management of educational systems (Seidel, 1969). In Figure 1, it is implied that mainline applications will be more cost effective than adjunct but will require higher capital investment in hardware and instructional design and development. Also, it will require a self-paced, individualized scheduling system and an objective-oriented conception of grading (Bunderson, 1970).

Adjunct applications, on the other hand, will often be more innovative than a total system engineered and produced for wide distribution. The individual faculty member and his students, especially in higher education, can use the computer in modeling, rethinking, and restructuring a portion of their field. This restructuring can have a major influence on the design of the next generation mainline program in that subject. Furthermore, giving the student skill in problem solving in his field can both motivate him and enable him to explore and discover far beyond the bounds of a particular course. It is no wonder that the adjunct mode is catching on so rapidly in the United States and has so many enthusiasts. It leaves the teacher in control, but it can still open up new worlds for many of the students. Unfortunately, enthusiasm for the adjunct mode seems to make some feel an obligation to attack other uses of the computer.

Good teachers immersed in teaching and research not only have the potential for restructuring their subject matter in the adjunct mode, but regard with suspicion and jealousy any encroachments on their domain by "outsiders." The good college professor's understanding of the beauties and subtleties of his field, his scorn for clichés and pedestrian approaches, and his treasured intuitions about pedagogy make him a "tough customer" to sell on the idea that he could profit from the contributions of an instructional designer.

One who takes the position that the faculty author alone can develop CAI programs, with no special training other than in a simple but powerful author language (TUTOR) is Dr. Don Bitzer at The University of Illinois. According to Bitzer (1970), in CAI, "The author needs no middleman." Yet at The University of Texas and at other CAI Laboratories and Centers, we are trying to make instructional development a team matter through introducing a person or persons trained in instructional software design into the authoring loop. There is an analogue to the author-editor relationship in publishing, but the instructional designer has a more intensive interaction with the author at each stage of development than does a textbook editor, and he has a much greater influence on the form and structure of the resulting product.

The differences of opinion may melt away when the distinction between adjunct and mainline is made. Bitzer can make his statement stand up when the product is an adjunct program, but it is less effective as we move along the continuum toward a mainline system. He can back up his statement with experiences of authors using his fine TUTOR language and PLATO system, but he cannot back it up from experiences at other centers

using languages like Coursewriter and less flexible graphics terminals. What this means is that his authors do not use keypunch operators or specifically assigned programmers or coders between them and the system. He does not speak specifically to the need for an instructional software designer as defined here.

When I assume the burden of proof for the necessity of an instructional designer in CAI authoring, that burden is a great one, for I must admit that the author can do without him. With equal logic, the eighteenth century farmer could prove that his field of corn would grow without the agricultural agent's help.

Figure 2 places the contributions of instructional software design in context. The three theoretical probability distributions represent the probability of obtaining a certain weighted quality rating in developing a CAI program:

a: by teachers with no middleman and no instructional design training.

b: by the same teachers on an instructional design team.

c: by a new generation of teachers trained in both their field and in instructional software design.

In Figure 2, it is assumed that a standard quality rating can be obtained based on learner performance, enjoyment, excellence of subject matter representation, and efficiency. It is also assumed that all CAI programs from adjunct to mainline, could be measured in this way, because instructional software engineering can improve the quality of adjunct programs as well.

Observe from Figure 2 why it is difficult to make a case for instructional software engineering. The existence of one or two high quality programs developed by lone, talented teachers is more visible than (for all practical purposes) the unmeasurable mean difference between distributions *a* and *b*. Despite this, that difference between *a* and *b* could be of incalculable value to education.

Insert Figure 2 about here

Instructional Design Products and their Uses

To present the case for instructional software design, it is useful to describe the products of this approach, although it is beyond the scope of this paper to describe in detail the processes and techniques of Instructional Software Engineering. Some relevant references are Bunderson (in press), Mager and Beach (1967), and Eraut (1967).

In general, instructional software design has the flavor of systems engineering. That is, the context of the course to be developed in a larger system is considered; the course is considered as a "black box" with definite and measurable input and output in terms of student performance; the black box is analyzed into component black boxes; a mock-up is synthesized and tested against its output specifications; and the feedback from testing is used for revision until the system performs as specified.

A systematic procedure for instructional design and development incorporates these system engineering concepts in a system that also provides for management and quality control of program development. The operation of these concepts can be inferred from a review of the products of this systematic approach. These products may be classified as public documents, intermediate design products, or final program materials.

Insert Figure 3 about here

The "author with no middleman" will usually produce only the digital code and perhaps the slides, tapes, or booklets which accompany the final program. He may provide minimal technical specifications in terms of a computer listing. This listing, however, may use type-codes which do not correspond to CRT characters, and in other ways be complex and difficult to decipher. This minimal documentation may allow him to exchange small adjunct programs with friends or colleagues, but not mainline programs.

When people ask such an author, "What does it teach?", he is at a loss without written statements of goals and objectives. If they ask, "Does it succeed?", he needs a description of the target population and, better, a prerequisite's test. If asked for a copy of the program, he is at a loss without user documentation and technical specifications for the programmer who will maintain it. Clearly, these problems increase in severity exponentially as we depart from small adjunct programs and talk about complex mainline systems.

From the "intermediate design products" the structure of a systematic approach to instructional development can be inferred. These products consist of notes, prose passages, flowcharts, manuscripts, student data, and other ephemeral or rapidly changing forms of information. They result from a sequence of important design decisions.

The three overall aspects of the systems engineering approach can be seen in the list of intermediate products. Context is considered through the needs, goals, and justification which result in "brochure information" useful for potential users or as part of a development proposal. In the box are listed those design products which arise in connection with the synthesis of the "black box." Performance objectives which lead to criterion test and prerequisite test define the input-output specifications. (Other specifications in the form of constraints, such as time, may also be determined.) The analysis of objectives and definition of the system architecture in terms of a hierarchy or other structure of intermediate objectives is the key step in this process. Synthesis of mechanisms for individualization and representational conventions for display and response for each subordinate objective depend on the analysis step. The special training of the instructional designer is most critical in the stage of design indicated within the box in Figure 3.

The notion of testing and iterative revision is implicit in the concept of formative evaluation (Scriven, 1967). This is more interesting than summative evaluation for the empirically oriented designer, for it can be characterized as a continuing cycle between experiment and adjustment until the

program seems to be working. Summative evaluation is most relevant to the production of brochure information and professional publications--to convince others that the program works. It is also useful to obtain field logistics data (distributions of completion times, housekeeping details, etc.) for the user manual.

The concept of formative evaluation provides a strong answer to those critics of mainline programming efforts who have often told me, "It is inappropriate to undertake these projects until we know ____________." (The blank may be filled in by "how people learn," "what reinforcement to give different students," "what instructional strategy to use for different students," etc.) Happily, we can proceed through, at worst, a combination of rank empiricism (to identify deficiencies) and the use of intuition and common sense to revise it until it does achieve its objectives. We naturally wish to base our revisions on instructional theory as fast as it is developed. It may be the case that the existence of high quality CAI courses with all displays, responses, and sequences operationally defined may be a prerequisite for its development, however.

The main concept in the first column of Figure 3 is that proper documentation for CAI programs cannot be determined until it is recognized that there are different audiences for documentation. The potential user needs brochure information, especially the institutional need, which describes a real problem in a real institutional setting that generated the program development. The justification for using CAI to meet this problem is most crucial to the potential user. He also needs an overview of how the program works, a review of its coverage (goals) and objectives, a definition of the target population, and any validation and cost data

available. Much of this same information, plus a description of societal needs and a production plan for all products, is needed by a funding agency.

Design architecture and rationale are of interest to sophisticated potential users, but full detail is most appropriate for professional publications. The pressure on universities in the United States from state legislatures to concentrate on teaching undergraduate students is in conflict with the "publish or perish" research ethic. A possible rapprochement is through doing research on the structure, organization, and pedagogical logic of one's discipline in the context of applied curriculum development projects. Such research may lead to important simplifications and reconceptualizations which may actually represent a theoretical contribution to that discipline. For example, Kekule's invention of the benzene ring representation simplified an array of complex phenomena for students as well as for chemists. Some analysis of subject matter undertaken in connection with CAI development has uncovered ambiguities and led to clarifying research.

Other audiences who need special forms of documentation include technical personnel who will operate, maintain, and update a complex mainline CAI program, managers, teachers, and proctors who will administer it, and students who will take it.

Summary of Instructional Design Contributions

The preceding brief review of the products of systematic instructional software design provides a means of identifying what the lone faculty author will probably fail to achieve unless he joins a design and production team. He can gain from employing much of this systematic approach even if he is working only on an adjunct CAI application. It is most unlikely that he

could or would want to produce by himself a mainline CAI system suitable for dissemination. In general, he would probably fail in the following:

a: He would fail to consider fully the "total system" aspects of the program to provide for wide dissemination. He would not likely consider documentation nor justification, nor have a management plan for development, nor heuristics for quality control during the development.

b: He is most unlikely to have skills in deriving performance objectives from goals and performing a behavioral analysis of objectives. Performance objectives are the operational definitions of sub-system output which make the system testable and improvable. A well stated objective includes both "what is given" and "what is performed," so that it leads to the specification of conventions for display and response in a rational way.

c: While with well designed software for handling student data it is possible to give the lone faculty author excellent feedback from student runs for revision, well designed summative validation studies are less likely to be produced. Research or theoretical publications which arise from the design architecture and rationale activities implied by Figure 3 are also less likely to result from a lone subject-matter expert. He needs inter-disciplinary stimulation from those familiar with the concepts of measurement, research design, and the power of changes in representation to reformulate problem areas.

Concluding Thoughts: First Issue

Discussions in the United States over who shall author CAI programs are headed in part by the fear of teachers that they will lose control over the content of instruction or perhaps even be replaced. Another problem has been the confusion between adjunct and mainline

applications of CAI. Using a team concept for instructional development enables us to leave control over the content of curriculum in the hands of talented teacher-authors who will learn to write for CAI systems. These will have no more control than the textbook writers now have. Instructional designers working with these authors will have a profound effect on the structure and organization of curricula, but they will not establish goals and objectives independent of the faculty authors.

The instructional design team will function primarily in larger projects designed to produce mainline systems, although consultation in various aspects of design and evaluation can be provided to independent faculty authors. The creative use of the computer in adjunct applications, using languages like APL, should be encouraged. Teachers and students can, with little effort, generate clever and useful simulation, drill, and other modules which improve the quality of classroom or laboratory instruction. Some of these modules will later influence or be incorporated into mainline systems.

"Programmed Instruction" vs Exploratory Approaches

The second issue which divides thinking in the United States is a reaction in part against the restraining, boring aspects of programmed instruction, and in part against the great difficulty and cost of developing carefully programmed materials employing tutorial strategies heavily. It is argued that the computer's unique capabilities should be exploited, rather than turning it into a stodgy page turner or narrowminded drill master. The result has been increased emphasis on simulation, "discovery learning " from

programs modeling some aspect of the subject matter but not teaching didactically, and a search for generative systems to produce exercise material (Nelson, 1970; Papert, 1970)

Clearly, this is a multifaced issue which cannot be given justice in a short time, especially since I wish to present some empirical data. Let me focus therefore on two popular techniques, discovery learning and learner control, and try to show that one cannot escape the need for careful software design by adopting these approaches uncritically.

Most people whom I know that are in a position to work with CAI in higher education have doctor of philosophy degrees. They are usually quite bright. In their own student days, they stood out in their ability to discover answers to problems on their own initiative and to organize, schedule, and complete their own learning activities. It is not surprising that a large majority of them take warmly to the idea of letting the student control his own learning and encourage him to learn by discovery. It is also the case that CAI programs which provide less tutorial guidance for the multitude of possible student errors, and indeed do not consider the structure of prerequisite objectives, are far less difficult and expensive to prepare. If the target population for a CAI program consists only of highly selected, bright, inner-directed students, an author can get away with heavy emphasis on discovery and learner control. In the mass enrollment situations where CAI will have its greatest economic impact, it is doubtful that this will be the case.

Consider some of the results of a program of research we have undertaken at The University of Texas at Austin for the past several years. This research has dealt with discovery versus expository learning and

learner control. For the most part, this research has employed an imaginary "science" task, *The Science of Xenograde Systems*. A Xenograde system consists of a nucleus containing small particles called alphons. One or more satellites may revolve around the nucleus, also containing alphons (Figure 4). Under certain conditions, a satellite may collide with the nucleus, exchanging alphons and affecting satellite velocity. The student must learn an algorithm to calculate the status of the system in terms of alphon count, satellite distance, and other variables as a function of time. The task has the hierarchical structure of concepts and quantitative rules characteristic of many topics in science education. In addition, its imaginary content assures us that students are totally naive as to any of the concepts at the beginning of an experiment, so that we are dealing with new learning. Perhaps the greatest advantage of its imaginary character has been to enable us to concentrate on design variables--structure, display, etc., rather than subject matter variables.

Insert Figure 4 about here

Most of the studies reported here used science education or secondary education students, primarily juniors, enrolled in the College of Education at The University of Texas at Austin.

The first study used a simulation of a "Xenograde system recorder"--a device which is capable of recording in tabular form the states of a Xenograde system at discrete steps in time, given initial conditions. For all groups, a posttest was given to measure the extent to which students could calculate a Xenograde record themselves, given initial conditions. To

do this, they had to understand and use the simple linear functions and binary choices on which the simulation was based.

The simulation was programmed in Coursewriter II on the IBM 1500 instructional system using cathode ray tube (CRT) displays of instructions and Xenograde records. Students could input parameters and examine the Xenograde record generated by the simulation. The question we sought to answer was, "How should simulation be used in teaching new material of this sort?" A pilot study, involving four experimental groups, was set up to investigate this problem. These four groups are defined in Figure 5a.

Insert Figure 5a about here

Learning the Xenograde algorithms could readily be accomplished by learning to apply a sequence of 13 decision rules. Various degrees of structure could be provided to the student to assist him in learning these rules. The most structure was provided in the "Expository" group. For this group, the rules were presented in sequence, each on a separate page of a booklet. Parameters were displayed for the student to input to the simulation so that an example of that rule might be generated. Three test items were then given which required the application of that rule. If two out of three were passed, another rule was presented; if not, another example of the same rule, followed by three more test items, was presented.

As can be seen from Figure 5a, structure was taken away from each of the Groups II through IV until, in the "Raw Simulation" group, students were simply told to experiment with the simulation, generating their own

parameters until they understood all of the quantitative relationships necessary to generate any Xenograde record, given initial conditions.

The results of the pilot study made clear what we should have known in the beginning: Simulation alone is inappropriate for teaching totally new material. Students in the two simulation groups were extremely anxious, bewildered, and frustrated. By dint of prying information out of the experimenters and fellow students, and perseverance, some of them did learn as well as any student in the more structured situations, but it took them much longer, and some gave up.

While inappropriate for teaching new material, a simulation model may be used by a skilled instructor to illustrate complex relationships in context with much didactic instruction. A simulation may be used after basic concepts and principles are learned, to integrate them in the context of a meaningful problem. It could also be used for testing the acquisition of concepts and principles and the ability to use these in problem solving. It is also good for generating pedagogically useful examples and displays.

Our subsequent studies used only the "Expository" and "Discovery" groups. The program was revised and simplified, and the simulation aspect was removed, since , if the displays of examples are all known and selected for their value in clearly illustrating a rule, it is inefficient to have them generated by a slow algorithm written in Coursewriter II.

Previous research on discovery learning had indicated that it was usually less efficient than expository, but may aid retention and transfer to later learning. To investigate this in the context of new learning, we used

the Xenograde task with the first two treatment groups listed in Figure 5b. In addition to replacing the simulation-generated examples with fixed, pre-programmed examples, the statements of the rules were simplified and put on 16mm frames for computer-controlled display on the IBM 1512 image projector. The displays of pre-programmed examples were simplified by removing data irrelevant to the rule being illustrated with these data. After these revisions, students readily learned the task in both groups. The discovery group required significantly more examples, and hence time, to learn than the expository group. This mean difference is illustrated in Figure 6 by the *X*'s. As is our custom, we do not like to consider mean differences without an analysis of the individual differences which are concealed therein. Figure 6 therefore shows the linear regression lines of number of examples on reasoning ability, measured by means of separate tests, for both groups. In the expository group, reasoning ability makes no difference, but in the discovery group, it is the students low on reasoning who suffer, showing that discovery learning, while efficient and perhaps motivating for the brighter students, places a burden on other students which an expository treatment does not. The same pattern of regression lines appeared using associative memory ability as the covariable. Students low on memory had to take more examples in a discovery treatment but not in an expository treatment.

Insert Figure 5b about here

Insert Figure 6 about here

In spite of the greater exposure to examples in the discovery group, they did not do as well on the posttest as did the expository group. A mean difference in favor of the expository group was observed, and was found to be statistically significant. There were no significant mean differences between the two groups on a retention test taken two weeks later, nor on a transfer test that required the discovery of three new rules, given examples. This finding is contrary to the hypothesis favoring discovery learning for retention and transfer. While it is not safe to generalize too far, given the restricted nature of this task, its short duration (less than two hours), and the student population, we must recognize the real possibility that for *new learning*, a carefully programmed expository treatment will be of equal or greater effectiveness than a discovery approach, especially for the less able student. The carefully programmed expository approach will probably be more efficient.

Learner Control Studies

The prospect of learner control of the sequence of events in instruction is intriguing. Allowing students to control what they see and do next is a way of letting them ask questions without mastering the natural language interpretation problem in CAI. It puts the responsibility on the learner for organizing his own learning. There is the prospect that it will be more meaningful to him to receive instruction only after he has made an active decision to ask for it. Finally, by relieving the author of the necessity of being all-knowing with regard to what he should do next for a

particular student at a particular time, it could greatly reduce the development task in the areas of progress monitoring and automatic sequencing strategies.

Our first learner control study was concerned with the effects of learner control of sequence in new learning. The Xenograde task was correspondingly chosen as the experimental vehicle. A modification was made in the expository treatment to display a representation of the learning hierarchy showing the prerequisite structure of the 10 rules of the simplified task. The hierarchy is represented in Figure 7 as the student saw it. Four of the objectives seen by the student are also illustrated in Figure 7. The student could select any of the 10 "lessons" in any order, after which choice he received a rule on the slide projector, an example on the CRT, and then three test items. After studying these, he could select another of the 10 boxes in the hierarchy, including one studied previously. If he chose one seen before, he would see the same rule but a new example and new test items. He was required to select each box at least once.

Insert Figure 7 about here

The experiment, a doctoral dissertation in educational psychology by William P. Olivier* (1970), compared the performance of students in the learner control mode with students for whom the sequence was controlled by the program. For each student in the learner control group, a "yoked partner" in the program control group was assigned the identical sequence. To increase

*Now at the Ontario Institute for Studies in Education.

the statistical reliability of the results, additional students were assigned randomly to the program control group with different fixed sequences. These different sequences were related by an index to the hierarchical sequences. An index of 1.0 indicated strict conformance to the hierarchy, moving from bottom to top. An index of 0 represented a complete reversal, from top to bottom, while .5 represented a variety of sequences wherein as many subobjectives were taken after a higher level for which they were prerequisite as before.

The mean posttest results for the program control group as a function of the sequence index is shown by the solid line in Figure 8. You will note that as the conformance to hierarchical sequence was degraded, learning decreased, except when the sequence was completely reversed. By covarying inductive reasoning ability, we found that students in the scrambled sequence groups who were high on this ability were not hurt. They apparently were able to infer what they had missed in the skipped lessons. Students low on this ability did poorly when the sequence was scrambled.

This effect did not appear in the learner control group; in fact, an almost completely reverse effect occurred, as indicated by the dotted line in Figure 8. Because students in the learner control group selected themselves into a sequence category while students in the program control group were assigned randomly, we know of no way to treat these data statistically as a function of the sequence index. They are suggestive only. It may be that students who are willing to grapple with the task and "explore" it by looking ahead in an idiosyncratic fashion are more highly motivated, more interested, or more creative than students who are passively willing to select a regular sequence indicated by the author.

Insert Figure 8 about here

We were able to treat group means statistically, ignoring sequence. The program control group mean was significantly higher than the learner control group mean, in spite of the degrading effect of scrambling the hierarchical sequence for some.

The conclusion is very similar to the conclusion reached in the discovery learning studies: Except for a small number of exceptional students, learner control of the sequence of lessons in a hierarchy may be less effective for learning *new material* than a rationally-planned, carefully-designed program-controlled sequence.

We have recently completed a study of learner control of sequence in a program on the laws of exponents and scientific notation, logarithms, and dimensional analysis, using freshman math students (Judd, Bunderson, & Bessent, 1970). In this program most of the material had been encountered before in one form or the other and may constitute a review for a given student. Our hypothesis was that the student may be the best judge of what to skip over and what sequence to follow when the material is partially familiar. The data from this study present a complex picture, in some ways not unfavorable toward learner control. However, a group working under complete program control seemed to perform better than the other groups. This effect was not apparent for the higher-ability students, as measured by a pretest; but under certain conditions of learner control, students who did poorly on the pretest seemed to use certain of the learner control mechanisms to avoid learning.

Summary: Second Issue

Instructional software design need not produce CAI programs that look like traditional programmed instruction. It can and should exploit strategies that are natural to the computer and lead to the use of the computer by the student as a tool for exploration and discovery. The implications of the preceding data are clear, however: Program development strategies that simplify the author's task by leaving important information-processing burdens with the student are likely to pose no special difficulty for the brighter students, but they are likely to be both less effective and less efficient for students of average and below-average ability. In the United States the public funds necessary for the implementation and support of CAI are most likely to flow in the direction of programs for the economically and educationally disadvantaged, or remedial situations created by open enrollment, and for technical and vocational colleges. It is vain to believe that students in these situations can soon learn to profit from instructional strategies developed by the elite for the elite.

Instructional software design *can* succeed with these students, however, for it considers by careful analysis the hierarchical structure of prerequisite objectives which must be mastered. Thus, students can be started at the level of their ability and moved forward by carefully designed and thoroughly tested and revised modules to high levels of achievement. Some of these modules can employ, if appropriate, discovery, simulation, and learner control strategies. Among the high levels of achievement attained by careful design can be included the ability to profit from discovery approaches to learning and the ability to use learner control options wisely and efficiently.

Conclusions

Two issues about CAI were seen to be at the base of many current discussions and disagreements in the United States. One has to do with whether CAI is a new medium for the classroom teacher or whether it is part of a new technology which transcends the teacher and the classroom and redefines both. A second issue has to do with strategies employed in CAI programs. Should programs guide the students step-by-step through carefully analyzed sequences, or should the student use the computer to explore subject matter, solve problems, and discover concepts himself?

These two issues were viewed from the point of view of an emerging new discipline of "instructional software design," and in the context of a distinction between "adjunct" uses of the computer by teachers and students and "mainline" systems of individualized instruction involving CAI and humans in a new individualized organizational pattern.

An attempt was made to show that instructional software design could be used profitably by faculty members in the development of CAI programs as adjuncts to their classes, or by design and development teams for the preparation of computer-based instructional systems for wide distribution. In the former case, *instructional software design* was seen to be useful but not indispensible. In the latter case, it was seen to be almost indispensible.

As usual, differences on this issue exist because of different objectives of well-meaning people on both sides. Some wish to preserve and strengthen the role of the classroom teacher, making the computer another medium at his disposal. Others wish to transform the educational system and

make it more responsive to pressing social problems through the use of new forms of hardware, software, and management technologies. Those in the first group will be less resistent to adjunct forms of CAI than to mainline systems. Those in the second group will encourage both forms. They know that the educational system cannot be transformed to the modular, self-paced, individualized system they envision without carefully designed mainline systems. They also recognize that innovative approaches, motivation, and sometimes a restructuring of the content of a field comes when individual faculty members and students employ the computer freely in their disciplines. These innovative approaches can render some mainline systems obsolete and lead to new, simpler, and more powerful approaches to the design of systems to teach a discipline.

The second issue bears some resemblance to the first, in that those who favor the more exploratory, free use of the computer often tend to equate the products of a team curriculum development effort with a stereotype of boring, small-step programmed instruction or unimaginative tutorial CAI. Thus, they emphasize the free and innovative uses of the computer by individual faculty and students. Some data were presented bearing on the comparative effectiveness of simulation, discovery, and learner control approaches versus didactic approaches under program control. The data suggest that the less structured approaches are as effective and as efficient as the structured approach for the brighter students, but not for those of average or less than average ability. Thus, it was argued that the more easily prepared, less structured adjunct programs could not substitute for careful instructional design, especially for the poorer students.

The data show that students do learn from most of the unstructured approaches, however. The ability to learn by discovery and to control one's own learning may be educational objectives as important as those taught concerning any specific subject matter. It was suggested that through the application of instructional software design, instructional systems could be built to achieve efficiency and effectiveness for both kinds of objectives.

APPENDIX

Implications for Conference Participants

Europeans have the opportunity to profit from the experience of CAI researchers in the United States and indeed, to exceed the United States in productive use of CAI. The fact that you have had less money to spend on computers so far can be an advantage, since it encourages careful planning to centralize resources where there is the greatest likelihood of success. Unfortunately, in the United States the major CAI research centers have had to spend an inordinate amount of time getting funds to stay alive, while smaller projects come and go, eating up great amounts of capital. Federal funds for CAI have been distributed widely and shallowly, always subject to politics, so that many projects have failed to reach the critical mass or achieve the continuity necessary for productive work. The situation does not seem to be changing.

If you agree with me that instructionAL software design is necessary for the full development of CAI, then important consequences follow in your plans for introducing CAI.

The most important consequence has to do with staffing for curriculum development. At a university like Bari, it would be advisable to establish a center for instructional software development and research. A very modest center would have the staff outlined in Figure 9.

Insert Figure 9 about here

The center staff would provide management services. These would include the programming of the time-frame for completion of each product listed in Figure 3. In addition, the management necessary to allocate resources and to monitor progress would be provided. The center would also provide technical support services for computer operations and proctoring, especially those in support of formative evaluation. Support services for media production, printed materials, and programming consultation would also be provided. Most important, instructional design services would be provided, with an "authoring team" composed of author, instructional designer, graduate assistants or other helpers, and a programmer making up the project staff. Funds for subject-matter consultants would also be desirable.

The computer science and instructional psychology faculty who direct the center would, in their academic roles, teach graduate courses, supervise interdisciplinary doctoral students, conduct research, and in general, expand the borders of the field of instructional software design.

The ongoing interdisciplinary research in instructional design and theory would provide an ideal environment for the productive use of the computer by individual faculty members as well as design and development teams. These faculty and their students should be encouraged to develop adjunct applications to supplement their classroom and laboratory instruction. Here at Bari, most success in involving faculty and students will come through the use of the excellent APL language, which has proven its worth at our University and elsewhere in the United States. Given some promotional and training techniques which have been proven in the U.S.A., the center staff could soon stimulate much productive use of APL.

This is not true of the Coursewriter language, for it does not lend itself to easy use. To develop good Coursewriter programs requires the application of careful instructional design, a team approach, and an expensive, time-consuming development cycle. One way to enable rapid production of Coursewriter materials is through the use of preprocessors, such as those developed by Dr. Ed Adams and his colleagues at Yorktown Heights, New York, and the State University of New York at Stony Brook. A preprocessor developed by Dr. Peter Dean and Ralph Grubb at IBM, San Jose, California, is another important advance for Coursewriter users. One of the most worthwhile research efforts which could be undertaken in the interdisciplinary doctoral programs encouraged by the CAI center would be to develop other kinds of preprocessors and authoring systems for facilitating the instructional development process.

References

Adams, E. N. Field evaluation of the German CAI Laboratory. In R. Atkinson & H. A. Wilson (Eds.) *Computer-assisted instruction: a book of readings*. New York: Academic Press, 1969.

Bitzer, D. L., & Skaperdas, D. Some pedagogical and engineering design aspects of computer-based education. In Wayne Holtzman (Ed.) *Computer-assisted instruction, testing, and guidance*. New York: Harper & Row, in press.

Bunderson, C. V. Justifying CAI in mainline instruction. Paper presented at the National Science Foundation Sponsored Conference on Computers in Undergraduate Curriculum, University of Iowa, June, 1970.

Bunderson, C. V. The computer and instructional design. In Wayne Holtzman (Ed.) *Computer-assisted instruction, testing, and guidance*. New York: Harper & Row, in press.

Decision Maker: MIT's Professor Seymour Papert on tomorrow's education. *Computer Decisions*, 1970, 2(8), 6-7.

Eraut, Michael R. An instructional systems approach to course development. In Robert T. Filep (Ed.) *Teaching machines and programmed instruction: AV Communication Review*, 15(1), 1967.

Lekan, Helen A. (Ed.) Index to computer assisted instruction. Instructional Media Laboratory, University of Wisconsin, Milwaukee, 2nd ed., February, 1970.

Judd, W. A., Bunderson, C. V., & Bessent, E. W. An investigation of the effects of learner control in computer-assisted instruction prerequisite mathematics (MATHS). Computer-Assisted Instruction Laboratory, Technical Report No. 5, November 1970, The University of Texas at Austin.

Mager, Robert F., & Beach, Kenneth M., Jr. *Developing vocational instruction*. Palo Alto, Calif.: Fearon Publishers, 1967.

Morrison, H. W., & Adams, E. N. Pilot study of a CAI Laboratory in German. In R. Atkinson & H. A. Wilson (Eds.) *Computer-assisted instruction: a book of readings*. New York: Academic Press, 1969.

Nelson, Theodor H. No more teachers' dirty looks. *Computer Decisions*, 2(9), September 1970, 16-23.

Olivier, William P. Program sequence by ability interaction in learning a hierarchical task by computer-assisted instruction. Unpublished doctoral dissertation, University of Texas, Austin, 1970.

Scriven, Michael. The methodology of evaluation. In R. Tyler, R. Gagné, & M. Scriven (Ed.) *Perspectives of curriculum evaluation*. AERA Monograph Series in Curriculum Evaluation, Rand McNally, 1967.

Simon, Herbert A. *The sciences cf the artificial*. Cambridge: MIT Press, 1969.

Tuttle, John G. The historical development of computer capabilities which permitted the use of the computer as an educational medium in the United States from 1958 to 1968, with implications of trends. Unpublished doctoral dissertation, New York University, 1970.

Zinn, Karl L. Computer assistance for instruction. Center for Research on Learning and Teaching, University of Michigan, 1965. (mimeo)

Zinn, Karl L. An evaluative review of uses of computers in instruction. Project CLUE, Ann Arbor, U. S. Office of Education. (in press)

ADJUNCT → → → MAINLINE

--Context provided by teacher

--Programming by teacher and student or informal exchange.

--Represents an add-on cost.

--Requires low to moderate capital investment.

--Redesign of a complete man-machine system.

--Specifications and programming by design-production teams.

--Great economical potential: supplantive.

--Requires high capital investment.

Other Dimensions for Comparison

--Increased effectiveness: opportunity for restructuring objectives and subject matter.

--Modest but variable system requirements.

--Sometimes sophisticated graphics:
TTY's or typewriters,
Batch or interactive,
Standard languages.

--Fits with standard credit-hour scheduling.

--Increased effectiveness and efficiency.

--Specific engineering design:
CRT or plasma display,
Interactive and efficient,
Special author & student languages.

--Requires self-paced scheduling and grading.

Figure 1.--*A Useful Distinction Between Two Classes of CAI Programs.*

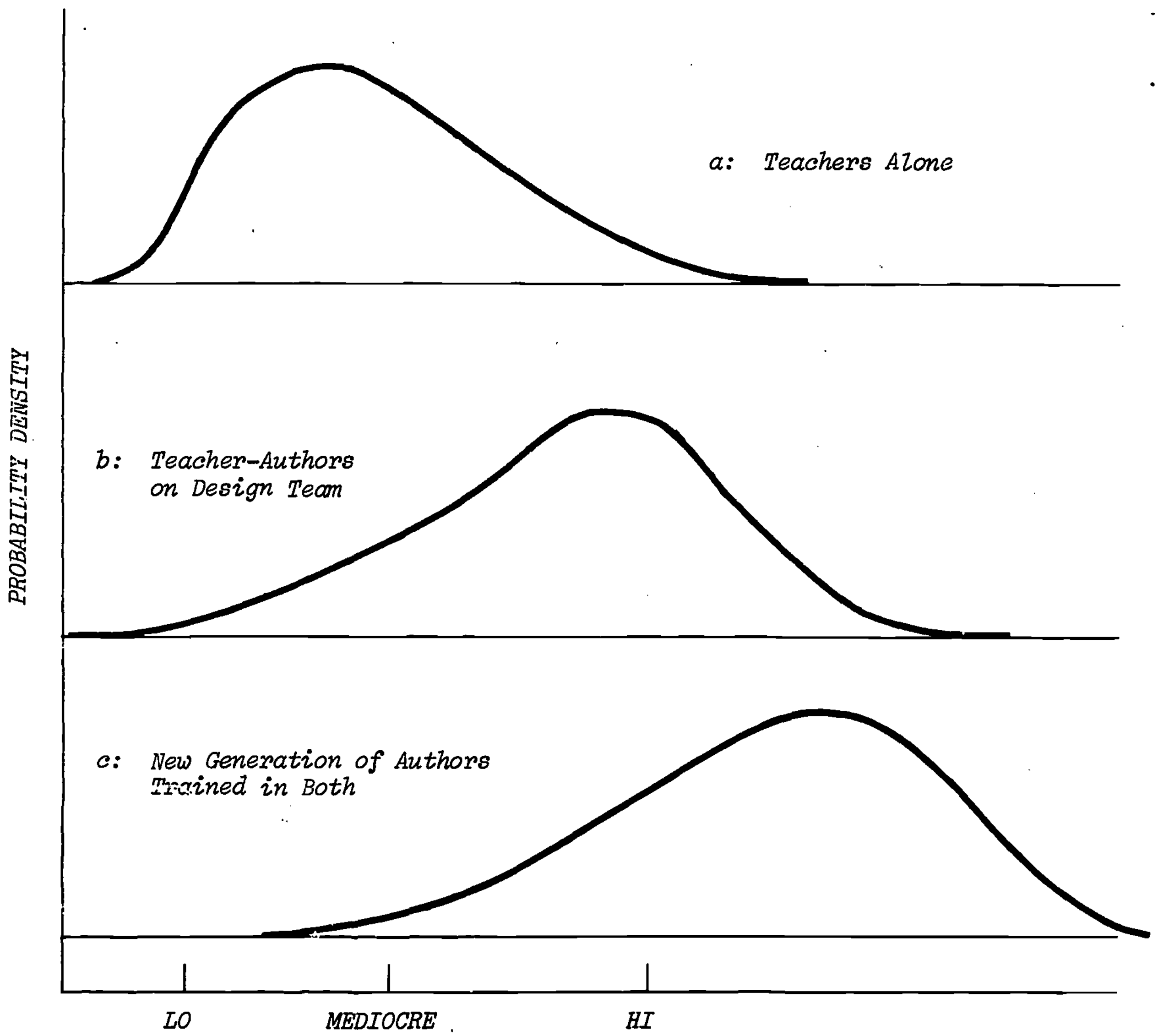


Figure 2.--*Theoretical Probability Distributions on Overall Quality of CAI Programs Developed by Different Procedures.*

Public Documentation | *Intermediate Products* | *Program Materials*

Brochure

Proposal

Professional Publications

User Manual

Proctor and Student Guides

I. Context information:

--Societal needs.
--Institutional needs.
--Goals: "Mastery Model"; prerequisites
--General description of approach; justification.
--Some evaluative data.
--Production plan (see V).

II. Design architecture & rationale:

--Performance objectives.
--Analysis; objectives and learning hierarchy.
--Synthesis; course structure and restrictions.
.Individualizing mechanisms, (flowcharts).
.Tests to measure objectives.
.Specification of display and response conventions, each subordinate objective.

--Technical evaluation and research reports.
.Formative (revision data).
.Summative (effectiveness and logistics data).

III. Manuscript or author's draft:

--Program steps and step formats; subroutines (for production personnel).

IV. Technical documentation; final program components:

--Program documentation for systems programmers.
--Documentation for operations: operator and proctor guides.
--Student manuals.

V. Production management plans for the production of all procedures listed above.

Prerequisite Test

Criterion Test

Diagnostic Test (if any)

Program:
Digital
A-V
Text

Figure 3.--*Products of Instructional Design.*

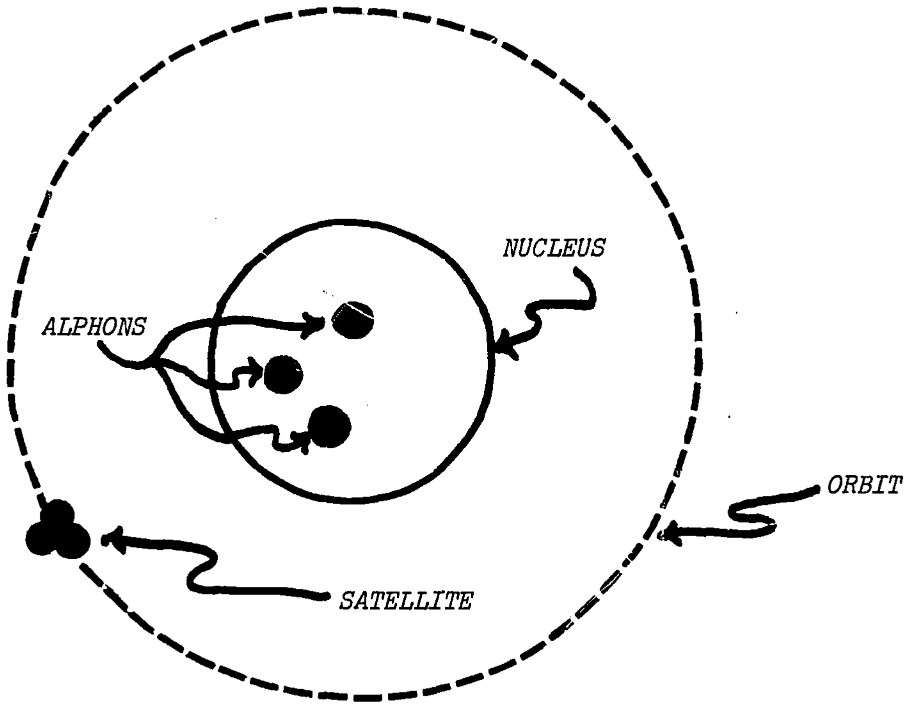


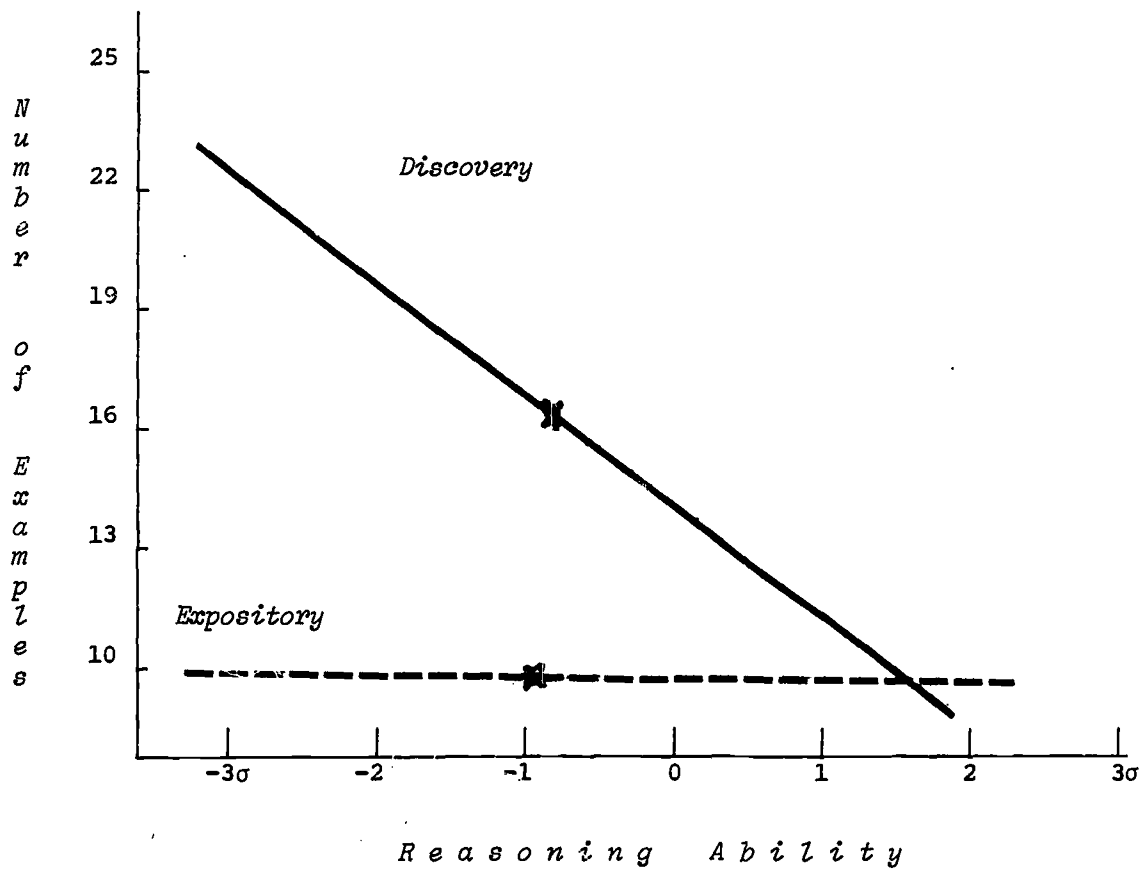
Figure 4.--*One way of characterizing a Xenograde System.*

Groups		*Written Rules*	*Example Parameters*	*Test Items*	*S's Own Parameters*
I:	Expository	X	X	X	X
II:	Discovery		X	X	
III:	Guided Simulation			X	X
IV:	Raw Simulation				X

Figure 5a.--*Treatment Options: First Xenograde Pilot Study*

Groups		*Rule on Slide*	*Simplified Example*	*Test Items*	*S Controls Sequence*
I:	Expository	X	X	X	
II:	Discovery		X	X	
III:	Learner Control	X	X	X	X

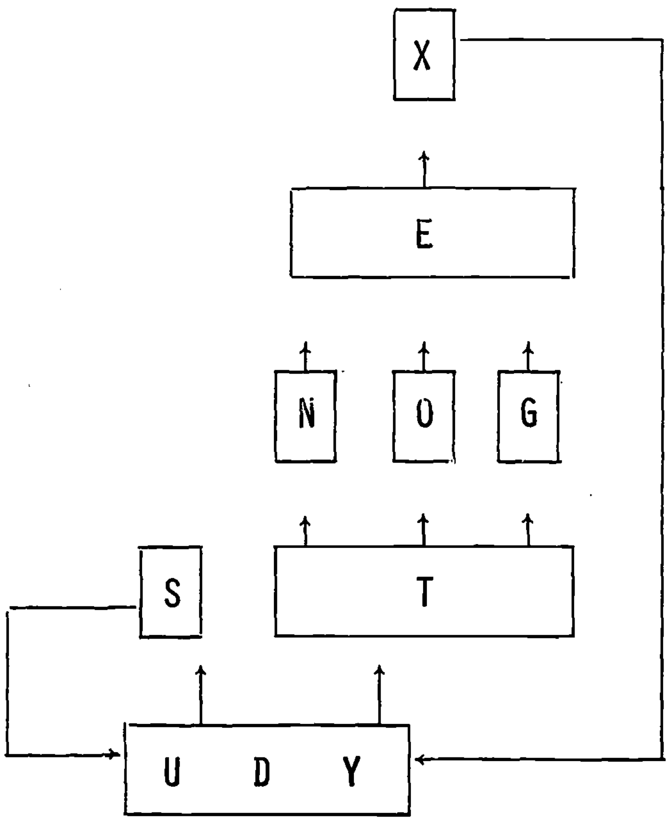
Figure 5b.--*Treatment Options: Subsequent Xenograde Studies*



Number of Examples
25
22
19
16
13
10
Discovery
Expository
-3σ
-2
-1
0
1
2
3σ
Reasoning Ability

Figure 6

HIGHEST LEVEL



LOWEST LEVEL

X: *Given the original satellite distance, the student should be able to predict to what maximum value the distance will increase.*

E: *Given that a blip has occurred, the student should be able to predict how the distance will begin changing.*

S: *Given that a blip has occurred, the student should be able to give the time of its occurrence and the value of distance at this time.*

U: *Given a previous distance, the student should be able to predict how FF and ACS will affect the values of distance.*

Figure 7.--*Diagram of the hierarchy of objectives for the Xenograde Science, along with four of the ten objectives also seen by the students.*

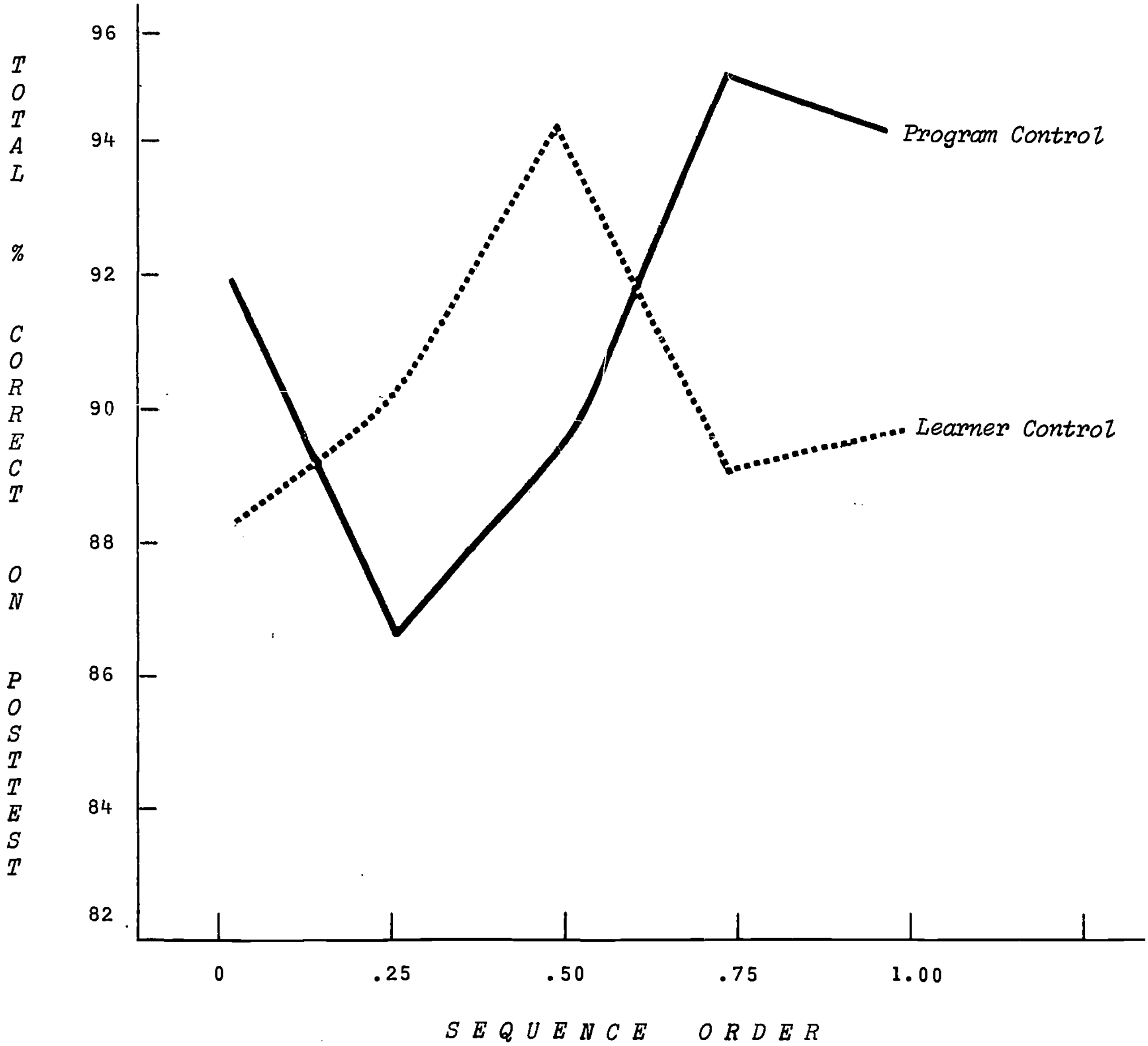


Figure 8.--*Performance as a Function of Sequence and Control Source*

*MANAGEMENT**

PhD Level

--One computer scientist and one instructional psychologist who teach half-time and devote half-time to research and development in instructional software design.

Other

--One full-time production manager for 5 to 10 curriculum development projects.

--Three full-time programming consultants.

--Five full-time secretarial-business personnel.

--Three full-time proctors.

--Two media specialists.

--Two keypunch or keyboard-input clerks.

For Each Mainline Curriculum Development Project

--One author: half-time teaching, half-time development.

--Budget for subject-matter consultants.

--Instructional design editor: one-fourth time per project.

--Two graduate assistants, half-time each.

--One half-time programmer.

*It is assumed that the computer center is maintained and operated by the university, exclusive of the CAI center staff.

Figure 9.--*Staff for a Modest University CAI Research and Development Center.*